Amendments of the Claims

1-14. (Cancelled)

15. (Currently Amended) A method comprising:

a. initializing shared variables, the shared variables including a turn variable, a first status flag, and a second status flag;

b. reading the shared variables into a memory;

c. entering a processing queue;

d. determining if a yield count has expired, wherein the yielding count is the amount of time a first thread is to remain in the processing queue;

e. if the yield count has expired, then exiting the processing queue and the first thread relinquishes its remaining quantum time to another thread in the processing queue;

f. if the yield count has not expired, then for a contending process, determining if a concurrent process exists;

g. retrieving the second status flag and the turn variable from the memory, reading the second status flag into a first cache and reading the turn variable into ~~the~~ a second cache to determine if the concurrent process is executing a critical section of code, wherein the second cache is larger than the first cache;

h. if the concurrent process is not executing a critical section of code, then entering the critical section of code, and upon completing the critical section of code, resetting the first status flag, wherein resetting the first status flag comprises retrieving a reset value from a register; and[[;]]

i. if the concurrent process is executing the critical section of code, then repeating d through i.

16. (Currently Amended) The method of claim 15, further comprising re-entering the processing queue after a period of time ~~wherein the second cache is larger than the first cache~~.

17. (Currently Amended) The method of claim 16 ~~15~~, wherein the period of time is determined by an operating system scheduling algorithm ~~resetting the first status flag comprises retrieving a reset value from a register~~.

18-30. (Cancelled)

31. (New) A machine-readable storage medium having stored thereon data representing sequences of instructions, the sequences of instructions which, when executed by a processor, cause the processor to perform operations comprising:

a. initializing shared variables, the shared variables including a turn variable, a first status flag, and a second status flag;

b. reading the shared variables into a memory;

c. entering a processing queue;

d. determining if a yield count has expired, wherein the yielding count is the amount of time a first thread is to remain in the processing queue;

e. if the yield count has expired, then exiting the processing queue and the first thread relinquishes its remaining quantum time to another thread in the processing queue;

f. if the yield count has not expired, then for a contending process, determining if a concurrent process exists;

g. retrieving the second status flag and the turn variable from the memory, reading the second status flag into a first cache and reading the turn variable into a second cache to determine if the concurrent process is executing a critical section of code, wherein the second cache is larger than the first cache;

h. if the concurrent process is not executing a critical section of code, then entering the critical section of code, and upon completing the critical section of code, resetting the first status flag, wherein resetting the first status flag comprises retrieving a reset value from a register; and

i. if the concurrent process is executing the critical section of code, then repeating d through i.

32. (New) The machine-readable storage medium of claim 31, wherein the sequences of instructions when executed by the processor, further cause the processor to perform operations comprising: re-entering the processing queue after a period of time.

33. (New) The machine-readable storage medium of claim 32, wherein the period of time is determined by an operating system scheduling algorithm.

WILLIAM THOMSON
SUPERVISORY PATENT EXAMINER